Fast Image Matting with Good Quality

Yen-Chun Lin[1], Shang-En Tsai[2], Jui-Chi Chang[3]

[1,2]Department of Computer Science and Information Engineering, Chang Jung Christian University
Tainan 71101, Taiwan

[3] Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei 10672, Taiwan

[1]yclin747@gmail.com; [2]shining701@gmail.com

Abstract

Image matting is a technique to extract the foreground from an image. A new Web-based image matting tool named Fast Image Matting (FIM) is presented in this paper. To extract the foreground, the user paints roughly along the boundary between the foreground and the background. The computation begins when painting begins, and a partial result can be seen while the user is painting with the mouse, which acts as a brush. Compared to NIM, which is a similar predecessor, FIM not only improves the user interface for better foreground extraction quality, but also uses a different method to improve the processing speed and foreground quality. Experimental results show that FIM can perform faster and better than well-known non-online Adobe Photoshop and three online foreground extraction tools.

Keywords

Foreground Extraction, Image Matting, Web-based Tool

Introduction

Image matting is a technique to extract the foreground object from an image based on the user input. It can be classified into two kinds of method. One is natural image matting from a single image; the other is matting from multiple images. Natural image matting usually classifies every pixel of an image into one of three kinds: the foreground, the background, and the combination of the former two (Chuang et al. 2001; Gastal and Oliveira 2010; Levin et al. 2008; Li et al. 2004; Lin et al. 2012; Lin et al. 2013; Nayi 2012; Shahrian and Rajan 2012; Shahrian et al. 2013; Wang and Cohen 2007). The combination ratio of the foreground and the background is called alpha. In other words, a pixel I of an image can be expressed as a combination of the foreground F and the background B:

$$I = \alpha F + (1 - \alpha)B, \qquad (1)$$

where α is the alpha and $0 \leq \alpha \leq 1$. Natural image matting needs to obtain the alpha of every pixel in the image, or the alpha matte. When the alpha matte is obtained, the foreground object is readily available.

In additional to the image to process, natural image matting requires additional information to make foreground extraction possible. The information is often a trimap, which partitions the original image into three regions, definitely foreground, definitely background, and unknown regions. For any pixel in the unknown region, we usually need to estimate its F and B values before its α can be decided. This is done by utilizing the additional information to make some assumptions on F and B to conjecture their values for each pixel in the unknown region. Once one or more pairs of conjectured F and B are available, a method is required to find the corresponding α of the pixel. When the α of every pixel in the unknown region is decided, the alpha matte is obtained.

Matting from multiple images solves the problem using two different images. No other information such as the trimap is needed. The methods of this category use the knowledge of the difference of multiple images to find out the foreground (Smith and Blinn 1996; Qian and Sezan 1999; Sun et al. 2006). A survey of pervious matting methods can be found in (Lin et al. 2012; Wang and Cohen 2007).

NIM is the first Web-based image matting application (Lin et al. 2012). It does not need the trimap; instead, the user paints roughly along the edge of the foreground. Magic Wand (Aviary) and Smart Cutout (FotoFlexer) are also online applications for foreground extraction. However, they do not use matting techniques, and are cutout algorithms. Unlike matting tools, cutout algorithms do not compute α when trying to extract the foreground. Adobe Photoshop CS5 (Dayley and Dayley 2010) is a well-known commercial product. It is not an online application, but it uses the matting approach to extracting the foreground.

A website for evaluation of the quality of image matting has been set up (Rhemann et al. 2009). It provides images and corresponding trimaps for matting applications to use. The matting results can thus be compared. This approach of comparison is not suitable for Magic Wand, Smart Cutout, NIM, Photoshop, and our new tool because they do not use a trimap to compute the foreground.

We have implemented a natural image matting tool, called Fast Image Matting (FIM). FIM is used through a Web browser, and its operation is very similar to that of NIM. However, its user interface is more precise and user-friendly than that of NIM. In addition, new techniques are exploited to improve the matting quality and reduce the amount of time required.

The rest of this paper is organized as follows. Section 2 introduces FIM, including its user interface and internal techniques to compute the alpha matte; distinctions between NIM and FIM are also described. Section 3 gives experimental results to show the merits of FIM over four other foreground extraction tools in terms of matting quality and the amount of time required. Section 4 concludes this paper.

Image Matting Tool FIM

User interface

We first describe the user interface of FIM, or how the user operates, and related improvements on NIM. Some information about NIM is briefly reviewed because it helps the presentation of FIM and particularly helps to show the merits of FIM. Techniques not directly related to user interface are described in the next subsection.

Fig. 1(a) shows the image of a flower and green background. The mouse is used as a brush to paint clockwise along the edge of the white flower. FIM computes the alpha matte immediately after the user has started to paint; when the brush moves, pixels covered by the path are processed. Fig. 1(b) shows that the brush has been painting from the left side of the flower; to the left of the brush track is the background (in black), and to the right is the foreground (in white). Fig. 1(c) shows that painting and computing have completed and that the foreground has been filled with white color to indicate the area for extraction.

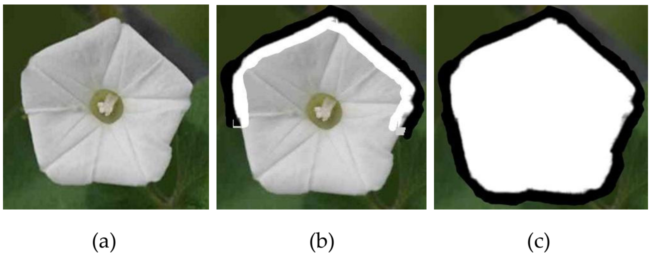


FIG. 1 EXTRACTING A FLOWER: (a) ORIGINAL IMAGE, (b) EDGE OF THE FLOWER BEING PAINTED CLOCKWISE AND HALF COMPLETED, (c) PAINTING COMPLETED AND FOREGROUND AREA FILLED WITH WHITE COLOR

Like NIM, while painting, the user of FIM can stop to change the width of brush; for example, when the edge of foreground changes from a smooth curve to a zigzag pattern, such as in the case of a shape consisting of thin, thread-like hairs, the brush should become larger to ease further painting and producing a quality foreground.

Although the user interface of FIM appears to be the same as that of NIM, FIM has two improvements. First, FIM removes some uncertainty for the user when painting along the boundary of the foreground object. Fig. 2 illustrates this difference between NIM and FIM. It shows the regions covered by the brush when NIM is used. The upward arrow indicates the direction of brush movement. The area painted by the brush is divided into five regions: definitely foreground, definitely background, unknown, decided foreground, and decided background regions. A pixel originally in the unknown region belongs to one of the two decided regions once its alpha value has been computed. However, the FIM brush covers only three regions; there are no definitely foreground and definitely background regions.

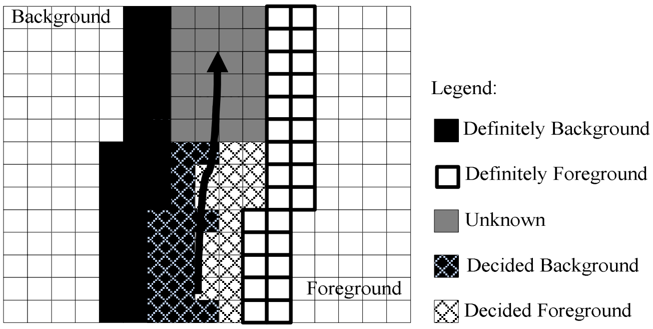


FIG. 2 THE REGIONS IN THE AREA PAINTED BY THE NIM BRUSH

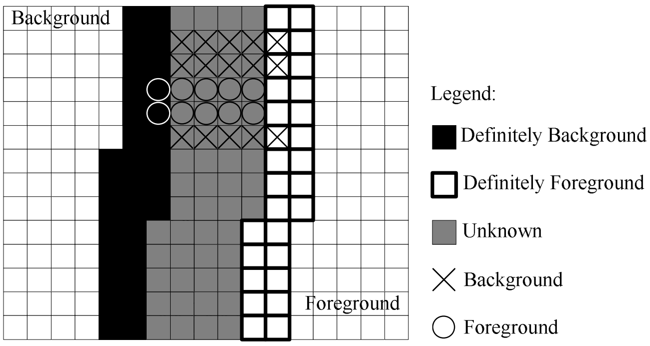
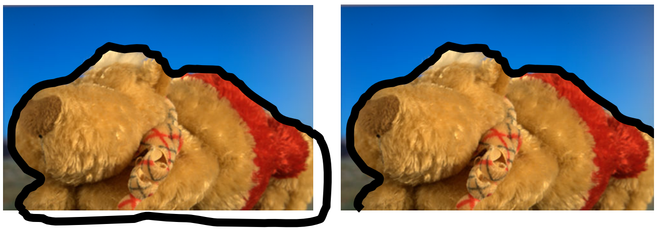


FIG. 3 SITUATION WHERE NIM MAKES ERRORS



(a) (b)

FIG. 4 TRACK OF BRUSH REQUIRED BY (a) NIM AND (b) FIM

Fig. 3 shows a situation where the use of five regions in NIM is not good. In this case, two foreground pixels (with a circle) are erroneously regarded as background because they are in the definitely background region, and three background pixels (with a cross) incorrectly become foreground since they are in the definitely foreground region. This may happen because the user does not know the exact widths of the definitely foreground and definitely background regions. Without these two regions, FIM never has this kind of errors.

The second improvement is for reducing the time required when the foreground object touches the edge of the image. For example, Fig. 4 (a) shows the track of the brush required by NIM, while Fig. 4(b) shows the track of the brush required by FIM. FIM can automatically achieve the same effect as NIM does in less time.

More new internal techniques

FIM takes a very different approach from NIM to obtaining the alpha value of pixels in the unknown region. NIM uses the time-consuming Gauss-Seidel iteration method to solve the global Poisson matting equation (Li et al. 2004). In contrast, FIM just uses the simple equation of (1), which implies that for each pixel I, if its F and B are known, its α can then be computed. FIM also uses a new, faster method to obtain F and B. To understand why FIM is faster than NIM, we will first present how NIM obtains foreground and background samples before explaining how FIM obtains foreground and background samples and computes alpha values.

Fig. 5 illustrates how NIM obtains F and B. The target pixel, of which the α value is being computed, is the one with oblique lines in the unknown region. To estimate the F value of the pixel, NIM uses the foreground window of the pixel, which is the smallest square that contains at least one definitely foreground pixel with the target pixel at the center. In Fig. 5, the window is a 3 × 3 square with red dotted sides, and contains two definitely foreground pixels (in white). The two pixels are foreground samples for estimating the F value of the target pixel.

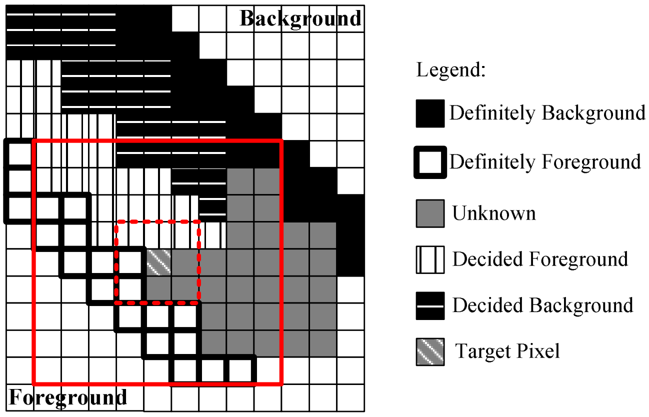


FIG. 5 TARGET PIXEL AND ITS FOREGROUND AND BACKGROUND WINDOWS WHEN NIM IS USED

The background window is the smallest square containing at least one definitely background pixel with the target pixel at the center. In Figure 5, it is a 9 × 9 square with red solid sides, and contains two definitely background pixels (in black). The two pixels are background samples for estimating B of the target pixel. As already mentioned, NIM then obtains the alpha matte by solving the time-consuming global Poisson matting equation.

FIM takes two stages to compute α. In the first stage, to reduce the amount of time consumed, FIM does not do it for every pixel in the unknown region. For nine neighboring pixels that form a square, FIM takes only the central pixel to compute for all nine pixels.

As illustrated in Fig. 6, the target pixel is the one with oblique lines in the unknown region. The foreground window is now the smallest square that contains at least one foreground pixel and has the target pixel at the center; similarly, the background window is the smallest square that contains at least one background pixel. To further reduce the processing time, in a foreground or background window, at most only eight pixels may be chosen as foreground or background samples. Fig. 6 shows that only three pixels with a number 3 on the right side are foreground samples, and only the two pixels with a red number 2 on the left side are background samples.

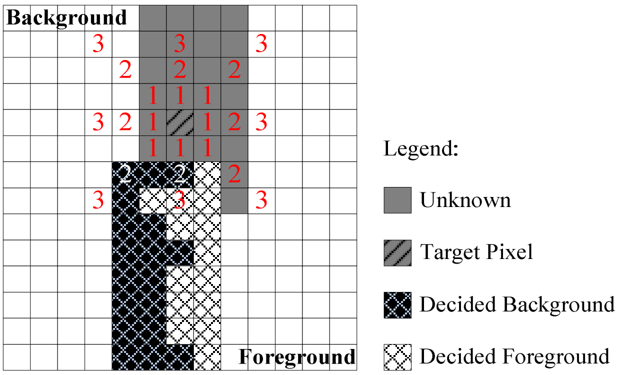


FIG. 6 AN ILLUSTRATION SHOWING FIM FOREGROUND AND BACKGROUND SAMPLES

In the second stage, unlike NIM, FIM does not use foreground and background samples to obtain a hopefully good estimation of F and B of the target pixel before α are decided; every pair consisting of a foreground sample and a background sample are used directly in the second stage to find the α values of the target pixel and its eight neighboring pixels.

FIM uses the RGB color values to decide the α value. The RGB color is a 3-D vector. Suppose the RGB value of the target pixel is $I = (I_r, I_g, I_b)$, the colors of m foreground samples are $F_i = (F_{ri}, F_{gi}, F_{bi})$, $i = 1, 2, \dots, m$, and the colors of n background samples are $B_i = (B_{rj}, B_{gj}, B_{bj})$, $j = 1, 2, \dots, n$. Thus, there are totally mn pairs of (F_i, B_j) samples. For every sample pair of (F_i, B_j), let α_{ij} be the ratio of the projection of I on vector $\overrightarrow{B_jF_i}$ to the length of $\overrightarrow{B_jF_i}$, or

$$\alpha_{ij} = \frac{\overrightarrow{B_jI}\cdot\overrightarrow{B_jF_i}}{\left\|\overrightarrow{B_jF_i}\right\|^2},$$

where $\left\|\overrightarrow{B_jF_i}\right\|$ is the length of $\overrightarrow{B_jF_i}$. Totally, there are at most mn values of α_{ij}, because an α_{ij} should be discarded if $\alpha_{ij} < 0$ or $\alpha_{ij} > 1$. Then, we compute the difference between the real color of the target pixel I and the color composed by Eq. (1) with the triplet (F_i, B_j, α_{ij}):

$$d_{ij} = \left\| I - (\alpha_{ij}F_i + (1-\alpha_{ij})B_j) \right\|.$$

At most mn values of d_{ij} are computed. A small d_{ij} means that the triplet (F_i, B_j, α_{ij}) can represent I well. Therefore, the α_{ij} of the triplet (F_i, B_j, α_{ij}) that generates the smallest d_{ij} is chosen as the α value of the target pixel.

We find that when the smallest d_{ij} is less than 0.5, the corresponding α_{ij} is satisfactory; otherwise, it is better to extend the foreground and background windows to have more samples to increase the chance of obtaining a d_{ij} less than 0.5.

As already mentioned, nine neighboring pixels can share the same foreground and foreground samples. Since each of the pixels may have its own color I, the same process in the second stage must be repeated to find the α value for each of the other pixels.

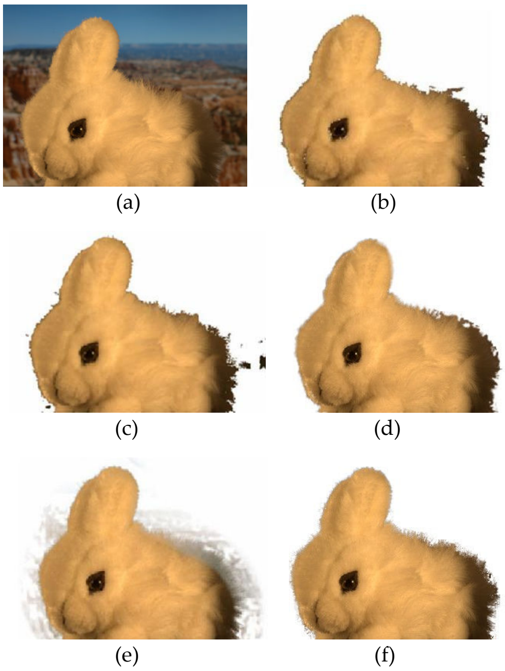
Among other less technical improvements, a new function of FIM is worthy of mentioning. When some foreground colors are very close to nearby background colors, a part of the foreground may be incorrectly classified as background. In this case, a tool, similar to an eraser, of FIM can easily enforce the correction by the user.

Experimental results

We have conducted experiments with many images to compare FIM with four other similar tools. Fig. 7 shows a typical result of foreground extraction by Magic Wand, Smart Cutout, NIM, Photoshop, and FIM, pasted onto a white background. As shown in the figure, the quality of foreground extracted by Magic Wand and Smart Cutout is generally poorer than that by NIM, Photoshop, and FIM. In general, Photoshop generates some "fog" around the real foreground object, and FIM can extract a foreground of better quality than the others.

Another typical example is shown in Fig. 8. NIM extracts a little darker color from the background. Photoshop generates some "fog" again. FIM appears to extract the best foreground.

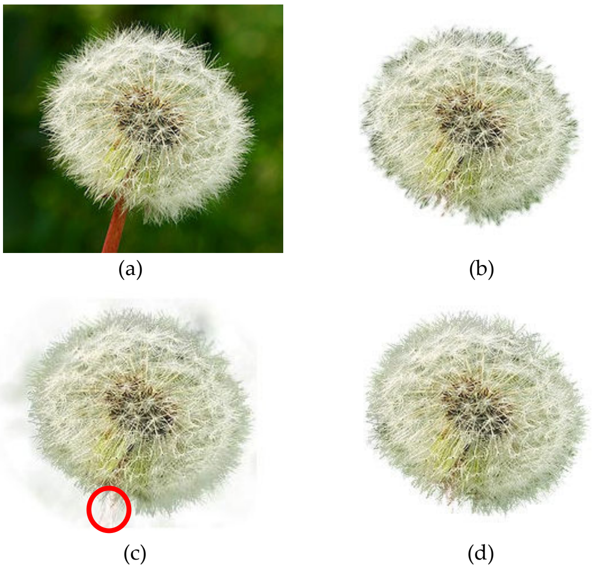
Fig. 9 gives an example showing the advantage of extending the foreground and background windows when the smallest d_{ij} is greater than 0.5. If the two windows are not extended, a few background pixels of darker color are incorrectly extracted. On the other hand, when extending the two windows is allowed, fewer background pixels, if any, are extracted.


(a) (b)

(c) (d)

(e) (f)

FIG. 7 (a) ORIGINAL RABBIT, AND EXTRACTED FOREGROUNDS BY (b) MAGIC WAND, (c) SMART CUTOUT, (d) NIM, (e) PHOTOSHOP, (f) FIM


(a) (b)

(c) (d)

FIG. 8 (a) ORIGINAL DANDELION CLOCK, AND EXTRACTED FOREGROUNDS BY (b) NIM, (c) PHOTOSHOP, (d) FIM

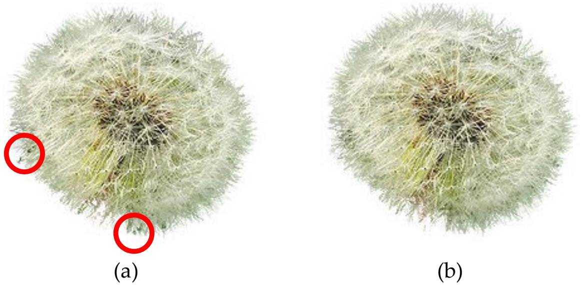

(a) (b)

FIG. 9 (a) EXTRACTED FOREGROUND WITHOUT EXTENDING THE TWO WINDOWS, (b) EXTRACTED FOREGROUND AFTER EXTENDING WINDOWS

FIM is not only better in terms of the quality of the extracted foreground object, but also faster than the other tools. Table 1 shows the amount of time required to extract foregrounds from five images. The amount of time is dependent on the size of images and the hardware platform used; for a given image, the experiments were conducted by using the same hardware and the same image size.

TABLE 1 THE AMOUNT OF TIME REQUIRED TO EXTRACT THE FOREGROUND (IN SECONDS)

	Magic Wand	Smart Cutout	NIM	Photoshop	FIM
Yellow flower	213	26	46	63	25
Dandelion clock	324	49	31	41	14
Gandalf	508	94	47	47	35
Rabbit	219	62	37	52	19
Dog	304	52	46	54	17

Conclusions

In this paper, we have presented FIM, an image matting tool for foreground extraction. Although it is similar to NIM, it improves the user interface and uses a very different method to obtain the alpha matte of an image. Experimental results show that FIM achieves better foreground quality and is faster than four other foreground extraction tools, including the well-known Photoshop.

ACKNOWLEDGMENT

This research was supported in part by Chang Jung Christian University.

REFERENCES

Aviary. "Web photo editor." Accessed July 17, 2013. http://aviary.com/web

Chuang Y.-Y., Curless B., Salesin D. H., and Szeliski R. "A Bayesian approach to digital matting." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2: 264-271, December 2001.

Dayley L. D. and Dayley B. "Adobe Photoshop CS5 Bible." Wiley, 2010.

FotoFlexer. "The world's most advanced online photo editor." Accessed July 17, 2013. http://fotoflexer.com/

Gastal E. S. L. and Oliveira M. M. "Shared sampling for real-time alpha matting." Computer Graphics Forum, 29(2): 575-584, May 2010.

Levin A., Lischinski D., and Weiss Y. ”A closed-form solution to natural image matting.” IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(2): 228-242, February 2008.

Li Y., Sun J., Tang C.-K., and Shum H.-Y. ”Lazy snapping.” ACM Transactions on Graphics. 23(3): 303-308, August 2004.

Lin Y.-C., Wang H.-A., and Hsieh Y.-F. ”Image matting through a Web browser.” Multimedia Tools and Applications, 61: 551-570, December 2012.

Lin Y.-C. and Yeh C.-H. "Improved online tool for image matting.” International Journal of Networked Computing and Advanced Information Management, 3(1): 16-24, April 2013.

Nayi N. G. ”Image matting for natural image.” International Journal of Engineering Research and Applications, 2(3): 2182-2185, May 2012.

Qian R. J. and Sezan I. M. ”Video background replacement without a blue screen.” Proceedings of the IEEE International Conference on Image Processing: 143-146, October 1999.

Rhemann C., Rother C., Wang J., Gelautz M., Kohli P., and Rott P. ”A perceptually motivated online benchmark for image matting.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: 1826-1833, 2009.

Shahrian E., Rajan D., Price B., and Cohen S. ”Improving image matting using comprehensive sampling sets.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: 636-642, June 2013.

Shahrian E. and Rajan D. ”Weighted color and texture sample selection for image matting.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: 718-725, June 2012.

Smith A. R. and Blinn J. F. ”Blue screen matting.” Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques: 259-268, August 1996.

Sun J., Li Y., Kang S. B., and Shum H.-Y. ”Flash matting.” ACM Transactions on Graphics, 25(3): 772-778, 2006.

Wang J. and Cohen M. F. ”Image and video matting: a survey.” Foundations and Trends in Computer Graphics and Vision, 3(2): 97-175, 2007.

Dr. Yen-Chun Lin joined the Department of Computer Science and Information Engineering, Chang Jung Christian University, Tainan, Taiwan as professor and chairman in August 2010, immediately after retiring from National Taiwan University of Science and Technology. In August 2010 he served as plenary speaker at 10th WSEAS International Conference on Applied Informatics and Communications, Taipei, Taiwan. In 2008 he served as plenary speaker at both the 7th WSEAS International Conference on Telecommunications and Informatics, Istanbul, Turkey and at the 7th WSEAS International Conference on E-Activities, Cairo, Egypt. He was Guest Editor of The Journal of Supercomputing in 2003; and Program Chair of the 2001 International Conference on Parallel and Distributed Computing, Applications, and Technologies. Dr. Lin received Honorable Mention of Annual Best Paper Award of Journal of Information Science and Engineering in 2001. He was a Visiting Scholar at the IBM Almaden Research Center, San Jose, California, from 1993 to 1994. His research interests include parallel computing, Web-based applications, and multimedia systems. Dr. Lin is an ACM Lifetime Member.

Dr. Shang-En Tsai received his PhD degree in Aeronautics and Astronautics from National Cheng Kung University, Taiwan in 2011. He was a member of the Structural Dynamics and Controls Lab (SDCL), University of Michigan, Ann Arbor from December 2009 to December 2010. He has been with Department of Computer Science and Information Engineering, Chang Jung Christian University, Tainan, Taiwan as assistant professor since August 2012. His research interests include network security and multimedia systems.